SQL Division

CS430/630
Lecture 7

Slides based on "Database Management Systems" 3rd ed, Ramakrishnan and Gehrke

Division

- Used to answer queries such as:
  *Find sailors who have reserved **all** boats.*
- Let *A* have 2 fields, *x* and *y*; *B* have only field *y*:
  - $A/B = \{\langle x \rangle \mid \exists \langle x, y \rangle \in A \ \forall \langle y \rangle \in B\}$
  - *A/B* contains **all *x* tuples** (sailors) such that **for every *y*** tuple (boat) in *B*, there is an ***xy* tuple in *A***
  - *Or*, if the set of *y* values (boats) associated with an *x* value (sailor) in *A* contains **all** *y* values in *B*, the *x* value is in *A/B*.
- In general, *x* and *y* can be any sets of fields (not singletons)

Examples of Division A/B

sid	bid
s1	b1
s1	b2
s1	b3
s1	b4
s2	b1
s2	b2
s3	b2
s4	b2
s4	b4

A

bid
b2

B1

bid
b2
b4

B2

bid
b1
b2
b4

B3

sid
s1
s2
s3
s4

A/B1

sid
s1
s4

A/B2

sid
s1

A/B3

Query 1

"Find the names of sailors who've reserved all boats"

$$\rho\ (Tempsids, (\pi_{sid,bid} \text{Reserves}) / (\pi_{bid} Boats))$$

$$\pi_{sname} (Tempsids \bowtie Sailors)$$

Query 2

"Find sailors who've reserved all red boats"

$$\rho(Temp,(\pi_{sid,bid} \text{Reserves})/(\pi_{bid}(\sigma_{color='red'} Boats)))$$

$$\pi_{sname}(Temp \bowtie Sailors)$$

Expressing A/B Using Basic Operators

- For *A/B*, compute all *x* values that are not **disqualified** by some *y* value in *B*
  - *x* value is **disqualified** if by attaching *y* value from *B*, we obtain an *xy* tuple that is not in *A*

Disqualified *x* values: $\pi_x((\pi_x(A) \times B) - A)$

A/B: $\pi_x(A)$ – **all disqualified tuples**

$$\pi_x(A) - \pi_x((\pi_x(A) \times B) - A)$$

Division in SQL

- Not supported as primitive operator
- Need to use nested queries to express division
  - One of the most subtle queries in SQL
  - Need to pay close attention to writing SQL division queries!
- There are two ways of writing division queries
  - Using the set **EXCEPT** operator (2-level nesting)
  - Without the **EXCEPT** operator (3-level nesting)

Division: Solution 1

"Find sailors who've reserved all boats."

With EXCEPT:

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS
        (
         (SELECT B.bid FROM Boats B)
         EXCEPT
         (SELECT R.bid FROM Reserves R
          WHERE R.sid=S.sid)
        )
```

Division: Solution 2

"Find sailors who've reserved all boats."

Without EXCEPT:

```
SELECT S.sname
FROM Sailors S                    Sailors S such that ...
WHERE NOT EXISTS (SELECT B.bid
there is no boat B ...            FROM Boats B
                  WHERE NOT EXISTS (SELECT *
  without a Reserves tuple showing S reserved B     FROM Reserves R
                                    WHERE R.bid=B.bid
                                      AND R.sid=S.sid))
```

"Find sailors who've reserved all **red** boats."

"Find sailors who've reserved all **red** boats."

With EXCEPT:

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS
        (
         (SELECT B.bid FROM Boats B
              WHERE B.color = 'red')
         EXCEPT
         (SELECT R.bid FROM Reserves R
          WHERE R.sid=S.sid)      )
```

"Find sailors who've reserved all **red** boats."

"Find sailors who've reserved all **red** boats."

Without EXCEPT:

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS (SELECT B.bid
                  FROM Boats B
                  WHERE B.color='red' AND
                        NOT EXISTS (SELECT *
                                    FROM Reserves R
                                    WHERE R.bid=B.bid
                                      AND R.sid=S.sid))
```

Another Example

```
Movies (movie_id, title,     year,          studio)
Actors (actor_id, name,      nationality)
StarsIn(actor_id, movie_id,  character)
```

"Find names of actors who star in ALL movies produced by Universal in year 1990."

```
SELECT A.name FROM Actors A
WHERE NOT EXISTS(

        SELECT M.movie_id FROM Movies M
        WHERE M.year=1990 AND M.studio='Universal'

        EXCEPT

        SELECT S.movie_id FROM Stars_In S
        WHERE S.actor_id=A.actor_id
)
```